EXHIBIT 9

Keinosuke Fukunaga

Introduction to Statistical Pattern Recognition

Second Edition

Introduction to Statistical Pattern Recognition

Second Edition

Keinosuke Fukunaga

School of Electrical Engineering
Purdue University
West Lafayette, Indiana

Morgan Kaufmann is an imprint of Academic Press

A Harcourt Science and Technology Company

San Diego San Francisco New York Boston
London Sydney Tokyo

This book is printed on acid-free paper. ♾

ACADEMIC PRESS
A Harcourt Science and Technology Company
525 B Street, Suite 1900, San Diego, CA 92101-4495 USA
http://www.academicpress.com

Academic Press
24-28 Oval Road, London NW1 7DX United Kingdom
http://www.hbuk/ap/

Morgan Kaufmann
340 Pine Street, Sixth Floor, San Francisco, CA 94104-3205
http://mkp.com

Library of Congress Cataloging-in-Publication Data
Fukunaga, Keinosuke.
Introduction to statistical pattern recognition / Keinosuke
Fukunaga. — 2nd ed.
p. cm.
Includes bibliographical references.
ISBN 0-12-269851-7
1. Pattern perception — Statistical methods. 2. Decision-making —
— Mathematical models. 3. Mathematical statistics. I. Title.
0327.F85 1990
006.4 — dc20 89-18195
CIP

PRINTED IN THE UNITED STATES OF AMERICA
03 9

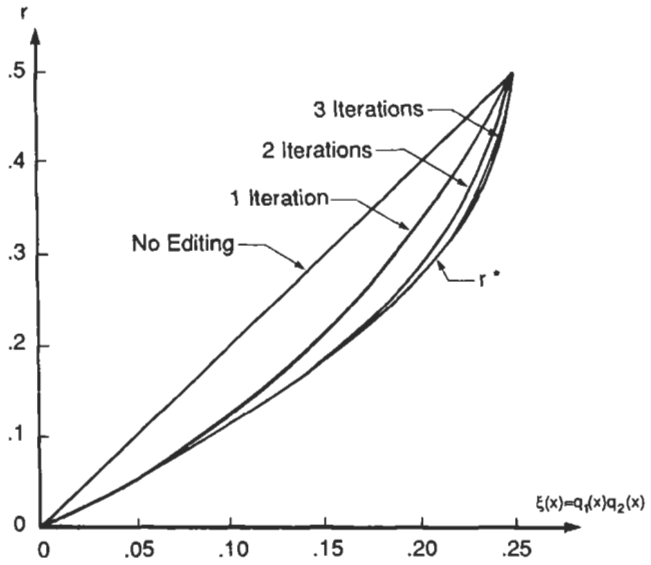

Fig. 7-16 Performance of repeated edited *NN* operations.

In this section, we touch briefly on some past efforts to overcome the above difficulty as follows.

Condensed *NN*: As seen in Fig. 7-3 for the voting *kNN* approach, the samples near the Bayes decision boundary are crucial to the *kNN* decision, but the samples far from the boundary do not affect the decision. Therefore, a systematic removal of these ineffective samples helps to reduce both computation time and storage requirements. This procedure is called the *condensed kNN decision rule* [23].

The risk value, $r(X)$, of each sample can be used as an indicator of how close the sample is located to the boundary. Therefore, we may set a threshold τ, and remove samples with $r(X) < \tau$. In addition, we had better remove all misclassified samples regardless of the value of $r(X)$, in order to avoid extra errors as was discussed in the edited *kNN* procedure. Since the effect of removing samples on the *kNN* error is hard to predict, it is suggested to classify test samples based on the condensed design set and confirm that the resulting error is close to the one based on the original design set.

Branch and bound procedure: The *kNN* computation time could be reduced significantly by applying the *branch and bound technique*, if design

samples can be hierarchically decomposed to subsets, sub-subsets and so on [24]. In order to describe the procedure, let us set up a tree structure as in Fig. 7-17 where each node represents a subset, S_k. Each subset is decomposed into

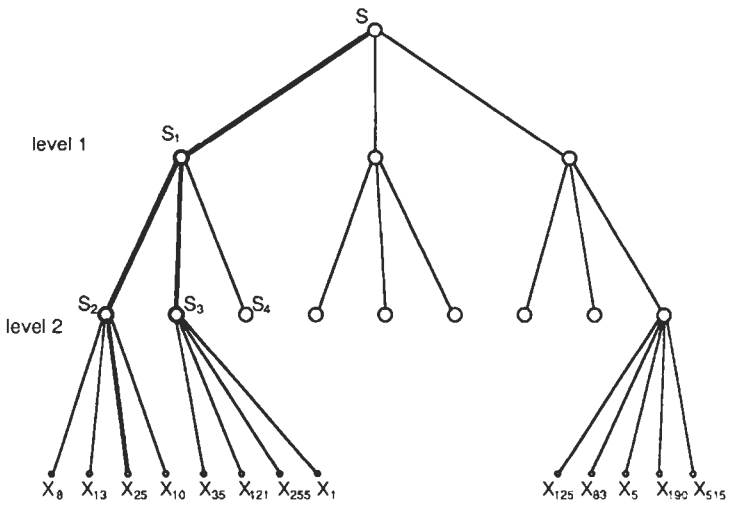

Fig. 7-17 A solution tree in the branch and bound procedure.

several other subsets, and at the bottom of the tree each node represents an individual sample. Each subset (or node) is characterized by the mean vector of the samples in S_k, M_k, and the furthest distance from M_k to a sample in S_k, d_k.

When the *NN* sample of an unknown X is sought, the search begins from the leftmost branch. After comparing the distances from X to X_8, X_{13}, X_{25}, and X_{10}, X_{25} is found as the closest one to X. The computer back-tracks the tree from S_2 to S_1 and then moves to S_3. However, before testing the members of S_3, it is tested whether or not the following inequality is satisfied.

$$d(X,M_3) > d(X,X_{25}) + d_3 \ . \tag{7.87}$$

If the inequality is satisfied, there is no possibility that the distance between X and any member of S_3 is smaller than $d(X,X_{25})$ [see Fig. 7-18]. Therefore, the search moves to S_4 without testing the members of S_3.

The procedure works well in a low-dimensional space. An experimental result for a uniform distribution in a two-dimensional space shows that only 46 distance computations are needed to find the *NN* among 1000 samples. In this

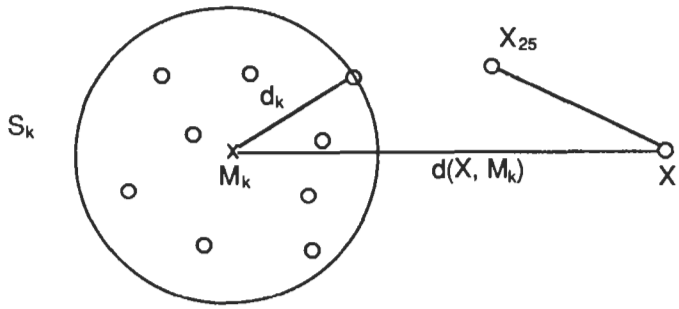

Fig. 7-18 A criterion to eliminate a group of samples.

experiment, the tree consists of 3 levels with each node decomposed to 3 nodes. At the bottom of the tree, there are 27 subsets containing 1000 samples. However, for an 8-dimensional uniform distribution, 451 distance computations are needed to find the *NN* from 3000 samples. The tree is formed with 4 levels and 4 decomposition, which yields 256 subsets at the bottom housing 3000 samples. As discussed in Chapter 6, all pairwise distances among samples become close, as the dimensionality gets high. Therefore, the effectiveness of (7.87) to eliminate subsets diminishes, and only a smaller number of subsets are rejected by satisfying (7.87).

Another problem of this method is how to divide samples into subsets. We will discuss this problem, which is called *clustering*, in Chapter 11. Again, finding clusters becomes more difficult, as the dimensionality goes up.

Computer Projects

1. Repeat Experiment 3 for Data I-Λ. Use (a) I and (b) $(I + \Lambda)/2$ as the metric.
2. Repeat Experiment 5.
3. Repeat Experiment 6.
4. Repeat Experiment 8.
5. Repeat Experiment 9.
6. Repeat Experiment 11.

Table 10-2 shows the number of subsets evaluated for three algorithms: two from the basic algorithm and one from the improved one.

TABLE 10-2

THE RESULTS OF THE BRANCH AND BOUND PROCEDURE

	NUMBER OF SUBSETS ENUMERATED			
Case	Exhaustive search	Basic algorithm without initial ordering	Basic algorithm with initial ordering	Improved algorithm
$\binom{12}{4}$	495	100	42	32
$\binom{24}{12}$	2,704,156	Did not terminate after 600 sec. CPU time (CDC 6500)	13,369 (249 sec. CPU time)	6,256 (140 sec. CPU time)

In the first algorithm, features are ordered according to the wave-lengths of the spectrum bands from the shortest to the longest, while in the second algorithm, features are ordered according to the class separability of individual feature. The basic algorithm was used for both cases. The third is the improved algorithm. Table 10-2 shows significant saving achieved by the basic algorithm with initial ordering and the improved algorithm [15].

Experiment 10: To evaluate the performance of the algorithms for a larger problem, an additional set of 12 features was generated by taking the square of 12 features of Experiment 9. The covariance matrices and the means were computed for the 24 feature set. It is to be expected that the resulting 24 feature set is very correlated, and there may be several subsets that yield very close values of the criterion.

Again, significant saving was achieved by the basic algorithm with initial ordering and the improved algorithm. The superiority of the improved algorithm is to be expected, because in the large variable problem the initial ordering scheme for the basis algorithm will not be very effective at higher levels. In fact, the basic algorithm without initial ordering did not even terminate after 600 seconds of computation (by CDC6500). Table 10-2 summarizes the results [15]. Table 10-3 gives the number of nodes expanded (subsets

TABLE 10-3

SUMMARY OF BEHAVIOR OF ALGORITHMS

Algorithm		LEVEL											
		1	2	3	4	5	6	7	8	9	10	11	12
Basic with ordering	No. of nodes enumerated	13	91	370	776	1376	2083	2961	3656	4185	2953	2248	771
	No. of nodes rejected	0	9	89	152	293	290	669	691	2325	1107	1718	746
Improved	No. of nodes enumerated	13	91	323	631	1091	1674	2024	1742	1242	910	347	188
	No. of nodes rejected	0	13	90	127	199	452	756	800	466	633	180	168

enumerated) at each level and the number of nodes for which the inequality (10.128) was satisfied (subsets rejected) at each level [15]. Because of the reordering of the features, every node that was rejected at each level of the improved algorithm results in a large number of suboptimal sequences being discovered. Hence, fewer nodes are enumerated overall in the improved algorithm than in the basic algorithm. The additional complexity of the improved algorithm appears justified in the light of its efficiency. Also, the improved algorithm is independent of the initial ordering of the features.

Computer Projects

1. Find the suboptimal linear features from Data *I*-Λ by maximizing the Bhattacharyya distance.

(a) Use the algorithm for μ_1 dominant.

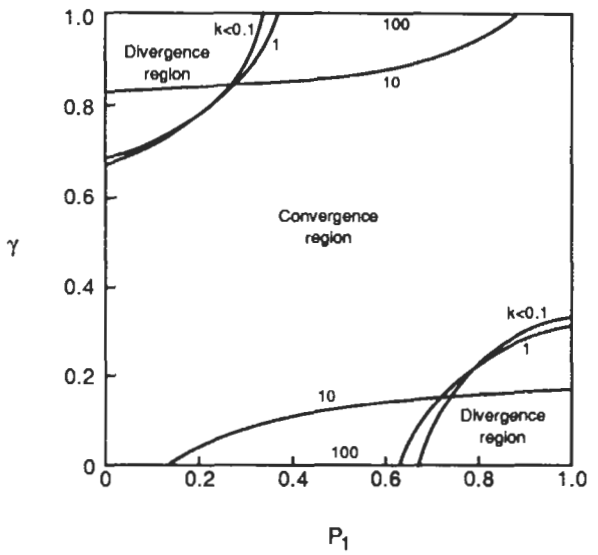

Fig. 11-5 Region of convergence.

Equations (11.23) and (11.24) with (11.34) show that we have three parameters in (11.34), ℓ_1/σ, ℓ_2/σ, and P_1 ($P_2 = 1 - P_1$), or

$$k = \frac{\ell_1+\ell_2}{\sigma}, \quad \gamma = \frac{\ell_1}{\ell_1+\ell_2}, \quad \text{and} \quad P_1. \tag{11.35}$$

In Fig. 11-5, the convergence regions of γ and P_1 are plotted for various values of k [5]. The figure shows that convergence is quite likely in practice, except for either extreme P_1's or γ's.

Branch and bound procedure [6]: The nearest mean reclassification algorithm works fine for many applications. However, there is no guarantee of the convergence of the iterative process. Also, the process might stop at a locally minimum point and fail to find the globally minimum point.

Since assigning a class to each sample is a combinatorial problem, the *branch and bound procedure* discussed in Chapter 10 may be applied to find the optimum class assignment.

Figure 11-6 shows a solution tree for the clustering problem with four samples and three clusters. In general, there are L^N different Ω's for classify-

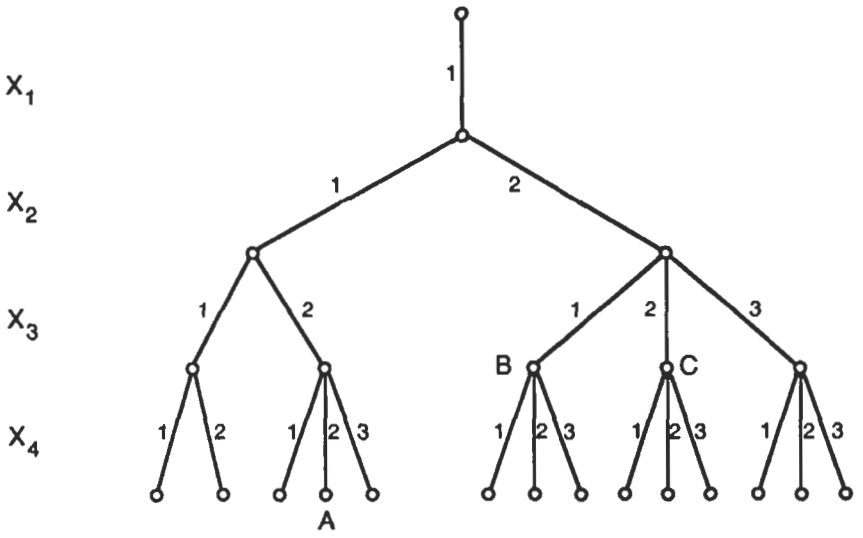

Fig. 11-6 A solution tree for clustering.

ing N sample vectors into L clusters. However, since the label of each cluster may be chosen arbitrarily, each classification could have several different expressions for Ω. For example, $\Omega = [1\ 1\ 2\ 2]$ and $\Omega = [2\ 2\ 3\ 3]$ are the same classification, both indicating that X_1 and X_2 are in one cluster and X_3 and X_4 are in one of the other clusters. In order to eliminate this duplication, we assign the first sample X_1 to cluster 1, the second sample X_2 to either cluster 1 or 2, and so on, as shown in Fig. 11-6.

In order for the branch and bound procedure to work effectively, we need to have a criterion which satisfies the *monotonicity condition*. Let $J_m(\Omega_m)$ be the criterion to be minimized for $\Omega_m = [\omega_{k_1} \ldots \omega_{k_m}]$, where the subscript m indicates the number of sample vectors involved. Then, the monotonicity condition is stated as

$$J_{m+1}(\Omega_m, \omega_{k_{m+1}}) \geq J_m(\Omega_m) \ . \tag{11.36}$$

That is, the J of a node is smaller than the J's for all nodes which are successors of the node.

Let α be the value of J_N which is the current smallest among all cases tested so far for the classification of N samples (for example, $\alpha = J(A)$). Then, the branch and bound procedure checks at each node (for example, at B) whether or not the following inequality is satisfied

$$J_m(\Omega_m) \geq \alpha \ . \tag{11.37}$$

If yes, then from (11.36), all successors of this node have J's larger than α. It means that the optimum classification does not exist in the subtree under the node. Thus, the subtree is rejected without further tests, and the search backtracks to the next node (for example, C). This elimination of subtrees makes the branch and bound procedure a very efficient tree search technique. When (11.37) is not satisfied, the search moves down to a node in the next level. The node selected for the next evaluation is determined by

$$J_{m+1}(\Omega_m, t) = \min_{i} J_{m+1}(\Omega_m, i) \ . \tag{11.38}$$

That is, X_{m+1} is assigned to cluster t, and the search goes on.

The criterion $J = \mathrm{tr}(S_m^{-1} S_w)$ satisfies the monotonicity condition with a minor modification. Again, assuming $S_m = I$, (11.11) is the criterion to be minimized. Since the number of samples is determined by the level of the solution tree and is independent of Ω, let us delete it from the criterion and rewrite the criterion for m samples, $X_1, \ldots, X_m$, as

$$J_m(\Omega_m) = \sum_{r=1}^{L} \sum_{j=1}^{m_r} \| X_j^{(r)} - M_r \|^2 \ , \tag{11.39}$$

where m_r is the number of ω_r-samples among $X_1, \ldots, X_m$. When X_{m+1} is added into cluster i, M_i must be modified to M_i', including the effect of X_{m+1}, and $\| X_{m+1} - M_i' \|^2$ must be added to the summation. Thus,

$$J_{m+1}(\Omega_m, i) = J_m(\Omega_m) + \Delta J(i) \ , \tag{11.40}$$

where

$$\Delta J(i) = \| X_{m+1} - M_i' \|^2 + \sum_{j=1}^{m_i} \{ \| X_j^{(i)} - M_i' \|^2 - \| X_j^{(i)} - M_i \|^2 \} \ . \tag{11.41}$$

The new i-class mean, M_i', can be obtained as

$$\begin{aligned} M_i' &= \frac{1}{m_i + 1} \left(\sum_{j=1}^{m_i} X_j^{(i)} + X_{m+1} \right) \\ &= M_i + \frac{1}{m_i + 1} (X_{m+1} - M_i) \ , \end{aligned} \tag{11.42}$$

or

$$X_j^{(i)} - M_i' = (X_j^{(i)} - M_i) - \frac{1}{m_i + 1}(X_{m+1} - M_i) . \tag{11.43}$$

Substituting (11.43) into (11.41) and using $M_i = (1/m_i)\Sigma_{j=1}^{m_i} X_j^{(i)}$,

$$\Delta J(\ell) = \frac{m_i}{m_i + 1}\|X_{m+1} - M_i\|^2 \geq 0 . \tag{11.44}$$

Since $\Delta J(\ell) \geq 0$, from (11.40) the criterion has the monotonicity property.

Note that (11.40), (11.44), and (11.42) provide recursive equations for computing $J_{m+1}(\Omega_m,\ell)$ and M_i' from $J_m(\Omega_m)$, M_i, and X_{m+1}. Also, (11.38), (11.40), and (11.44) indicate that the selection of the next node can be made by minimizing $\Delta J(\ell)$ with respect to ℓ.

For a large N, the number of nodes is huge. Thus, the initial selection of α becomes critical. One way of selecting a reasonably low initial α is to apply the iterative nearest mean reclassification algorithm to get a suboptimal solution and use the resulting criterion value as the initial α. The branch and bound procedure gives the final solution which is guaranteed to be optimum globally.

Also, it is possible to make the procedure more efficient by reordering the samples [6].

Normal Decomposition

Piecewise quadratic boundary: The nearest mean reclassification rule can be extended to include more complex boundaries such as quadratic ones. Following the same iterative process, the algorithm would be stated as follows:

(1) Choose an initial classification, $\Omega(0)$, and calculate $P_i(0)$, $M_i(0)$ and $\Sigma_i(0)$ $(i = 1, \ldots, L)$.

(2) Having calculated class probabilities, $P_i(\ell)$, and sample means and covariance matrices, $M_i(\ell)$ and $\Sigma_i(\ell)$, at the ℓth iteration, reclassify each X_j according to the smallest $(1/2)(X_j - M_i)^T \Sigma_i^{-1}(X_j - M_i) + (1/2) \ln|\Sigma_i| - \ln P_i$. The class probability for ω_i is estimated by the number of ω_i-samples divided by the total number of samples.

(3) If the classification of any X_j is changed, calculate the $P_i(\ell+1)$, $M_i(\ell+1)$ and $\Sigma_i(\ell+1)$ for the new class assignment, and repeat from Step (2).

Otherwise stop.

This process is identical to the nearest mean reclassification algorithm, but results in a piecewise quadratic boundary. Also, since the estimation of the covariance matrices is involved, the process is more computer-time consuming and more sensitive to parameters such as sample size, dimensionality, distributions, and so on.

More fundamentally, the clustering techniques mentioned above may have a serious shortcoming, particularly when a mixture distribution consists of several overlapping distributions. An important goal of finding clusters is to decompose a complex distribution into several normal-like distributions. If we could approximate a complex distribution by the summation of several normal distributions, it would be much easier to discuss all aspects of pattern recognition, including feature extraction, classifier design, and so on. However, the clustering procedures discussed above decompose a mixture as in Fig. 11-7(b) rather than as in Fig. 11-7(a). The hatched distribution of cluster 1 in Fig. 11-7(b) includes the tail of the ω_2-distribution and does not include the tail of the ω_1-distribution. As a result, the mean and covariance matrix of the hatched distribution in Fig. 11-7(b) could be significantly different from the ones for the hatched distribution of Fig. 11-7(a). Thus, the representation of a complex distribution by the parameters obtained from the clustering procedures described above could be very poor.

Decomposition of a distribution into a number of normal distributions has been studied extensively [7]. The two most common approaches are the *method of moments* and *maximum likelihood estimation*. In the former method, the parameters of normal distributions are estimated from the higher order moments of the mixture distribution (for example, the third and fourth order moments [8]). This approach is complex and not very reliable for high-dimensional cases. Therefore, in this chapter, only the latter approach is presented in detail.

Maximum likelihood estimate [9-10]: In order to obtain the hatched distribution of Fig. 11-7(a) from $p(X)$, it is necessary to impose a mathematical structure. Let us assume that $p(X)$ consists of L normal distributions as

Keinosuke Fukunaga

Introduction to Statistical Pattern Recognition

Second Edition

From reviews of the first edition:

"Contains an excellent review of the literature on information theory, automata, and pattern recognition."
—*Choice*

"Many topics are covered with dispatch. Also the book shows hints of the author's own careful and valuable research."
—*IEEE Transactions on Information Theory*

ISBN 0-12-269851-7

90051

9 780122 698514